SYSTEM AND METHOD FOR INTEGRATING ANIMATED GRAPHICAL ELEMENTS IN A SCROLLING DISPLAY WINDOW

A system and method for displaying animated graphical elements, representing static text, and static and animated graphical elements, in a scrolling ticker. The ticker elements are mapped to an off screen buffer (106) which is larger than the ticker window, and a portion of the buffer, beginning at an offset point (116), is repeatedly copied to the ticker window. Concurrently, any animated graphical element mapped to the buffer is updated, if needed, with the next frame (114). Ticker element objects may include headline and advertisement data retrieved from remote sources, and stored in separate headline and add channel files. The ticker may be scrolled either horizontally or vertically.
<table>
<thead>
<tr>
<th>Code</th>
<th>Country</th>
</tr>
</thead>
<tbody>
<tr>
<td>AL</td>
<td>Albania</td>
</tr>
<tr>
<td>AM</td>
<td>Armenia</td>
</tr>
<tr>
<td>AT</td>
<td>Austria</td>
</tr>
<tr>
<td>AU</td>
<td>Australia</td>
</tr>
<tr>
<td>AZ</td>
<td>Azerbaijan</td>
</tr>
<tr>
<td>BA</td>
<td>Bosnia and Herzegovina</td>
</tr>
<tr>
<td>BB</td>
<td>Barbados</td>
</tr>
<tr>
<td>BE</td>
<td>Belgium</td>
</tr>
<tr>
<td>BF</td>
<td>Burkina Faso</td>
</tr>
<tr>
<td>BG</td>
<td>Bulgaria</td>
</tr>
<tr>
<td>BJ</td>
<td>Benin</td>
</tr>
<tr>
<td>BR</td>
<td>Brazil</td>
</tr>
<tr>
<td>BY</td>
<td>Belarus</td>
</tr>
<tr>
<td>CA</td>
<td>Canada</td>
</tr>
<tr>
<td>CF</td>
<td>Central African Republic</td>
</tr>
<tr>
<td>CG</td>
<td>Congo</td>
</tr>
<tr>
<td>CH</td>
<td>Switzerland</td>
</tr>
<tr>
<td>CI</td>
<td>Côte d'Ivoire</td>
</tr>
<tr>
<td>CM</td>
<td>Cameroon</td>
</tr>
<tr>
<td>CN</td>
<td>China</td>
</tr>
<tr>
<td>CU</td>
<td>Cuba</td>
</tr>
<tr>
<td>CZ</td>
<td>Czech Republic</td>
</tr>
<tr>
<td>DE</td>
<td>Germany</td>
</tr>
<tr>
<td>DK</td>
<td>Denmark</td>
</tr>
<tr>
<td>EE</td>
<td>Estonia</td>
</tr>
<tr>
<td>ES</td>
<td>Spain</td>
</tr>
<tr>
<td>FI</td>
<td>Finland</td>
</tr>
<tr>
<td>FR</td>
<td>France</td>
</tr>
<tr>
<td>GA</td>
<td>Gabon</td>
</tr>
<tr>
<td>GB</td>
<td>United Kingdom</td>
</tr>
<tr>
<td>GE</td>
<td>Georgia</td>
</tr>
<tr>
<td>GH</td>
<td>Ghana</td>
</tr>
<tr>
<td>GN</td>
<td>Guinea</td>
</tr>
<tr>
<td>GR</td>
<td>Greece</td>
</tr>
<tr>
<td>HU</td>
<td>Hungary</td>
</tr>
<tr>
<td>IE</td>
<td>Ireland</td>
</tr>
<tr>
<td>IL</td>
<td>Israel</td>
</tr>
<tr>
<td>IS</td>
<td>Iceland</td>
</tr>
<tr>
<td>IT</td>
<td>Italy</td>
</tr>
<tr>
<td>JP</td>
<td>Japan</td>
</tr>
<tr>
<td>KE</td>
<td>Kenya</td>
</tr>
<tr>
<td>KG</td>
<td>Kyrgyzstan</td>
</tr>
<tr>
<td>KP</td>
<td>Democratic People's</td>
</tr>
<tr>
<td>KR</td>
<td>Republic of Korea</td>
</tr>
<tr>
<td>KZ</td>
<td>Kazakhstan</td>
</tr>
<tr>
<td>LC</td>
<td>Saint Lucia</td>
</tr>
<tr>
<td>LI</td>
<td>Liechtenstein</td>
</tr>
<tr>
<td>LK</td>
<td>Sri Lanka</td>
</tr>
<tr>
<td>LR</td>
<td>Liberia</td>
</tr>
<tr>
<td>LS</td>
<td>Lesotho</td>
</tr>
<tr>
<td>LT</td>
<td>Lithuania</td>
</tr>
<tr>
<td>LU</td>
<td>Luxembourg</td>
</tr>
<tr>
<td>LV</td>
<td>Latvia</td>
</tr>
<tr>
<td>MC</td>
<td>Monaco</td>
</tr>
<tr>
<td>MD</td>
<td>Republic of Moldova</td>
</tr>
<tr>
<td>MG</td>
<td>Madagascar</td>
</tr>
<tr>
<td>MK</td>
<td>The former Yugoslav</td>
</tr>
<tr>
<td>ML</td>
<td>Mali</td>
</tr>
<tr>
<td>MN</td>
<td>Mongolia</td>
</tr>
<tr>
<td>MR</td>
<td>Mauritania</td>
</tr>
<tr>
<td>MW</td>
<td>Malawi</td>
</tr>
<tr>
<td>MX</td>
<td>Mexico</td>
</tr>
<tr>
<td>NE</td>
<td>Niger</td>
</tr>
<tr>
<td>NL</td>
<td>Netherlands</td>
</tr>
<tr>
<td>NO</td>
<td>Norway</td>
</tr>
<tr>
<td>NZ</td>
<td>New Zealand</td>
</tr>
<tr>
<td>PL</td>
<td>Poland</td>
</tr>
<tr>
<td>PT</td>
<td>Portugal</td>
</tr>
<tr>
<td>RO</td>
<td>Romania</td>
</tr>
<tr>
<td>RU</td>
<td>Russian Federation</td>
</tr>
<tr>
<td>SD</td>
<td>Sudan</td>
</tr>
<tr>
<td>SE</td>
<td>Sweden</td>
</tr>
<tr>
<td>SG</td>
<td>Singapore</td>
</tr>
<tr>
<td>SI</td>
<td>Slovenia</td>
</tr>
<tr>
<td>SK</td>
<td>Slovakia</td>
</tr>
<tr>
<td>SN</td>
<td>Senegal</td>
</tr>
<tr>
<td>SZ</td>
<td>Swaziland</td>
</tr>
<tr>
<td>TD</td>
<td>Chad</td>
</tr>
<tr>
<td>TG</td>
<td>Togo</td>
</tr>
<tr>
<td>TJ</td>
<td>Tajikistan</td>
</tr>
<tr>
<td>TM</td>
<td>Turkmenistan</td>
</tr>
<tr>
<td>TR</td>
<td>Turkey</td>
</tr>
<tr>
<td>TT</td>
<td>Trinidad and Tobago</td>
</tr>
<tr>
<td>UA</td>
<td>Ukraine</td>
</tr>
<tr>
<td>UG</td>
<td>Uganda</td>
</tr>
<tr>
<td>US</td>
<td>United States of America</td>
</tr>
<tr>
<td>UZ</td>
<td>Uzbekistan</td>
</tr>
<tr>
<td>VN</td>
<td>Viet Nam</td>
</tr>
<tr>
<td>YU</td>
<td>Yugoslavia</td>
</tr>
<tr>
<td>ZW</td>
<td>Zimbabwe</td>
</tr>
</tbody>
</table>
SYSTEM AND METHOD FOR INTEGRATING ANIMATED GRAPHICAL ELEMENTS IN A SCROLLING DISPLAY WINDOW

RELATED APPLICATIONS:

The present application claims priority from U.S. Provisional Application Serial No. 60/099,570, entitled "SYSTEM AND METHOD FOR INTEGRATING ANIMATED GRAPHICAL ELEMENTS IN A SCROLLING DISPLAY WINDOW", filed on September 9, 1998, the entire contents of which are incorporated herein by reference.

BACKGROUND:

The demands of professional and personal goals, and lifestyles placed on an individual leaves little in the way of spare time and, as a result, many individuals merge the time spent between professional and personal activities at home and work. Technology enhances the ability to merge the lines between these activities. In general, people have a great need to maximize productivity to achieve their goals and often require access to various types of general and specific information to do so. The Internet puts a wealth of information at a user's fingertips. However, it can be a very cumbersome and time consuming environment. If an individual does not know the precise location of the information they seek, it can be difficult for them to find what they need, when they need it. Although various search engines are available, such tools often return a large number of results, many of which may be completely irrelevant.

In addition, due to the sheer volume of information available via the Internet, many users suffer from "information overload." When a user does locate a source on the Internet for information of interest, this location can be stored as a "bookmark" in an Internet browser. However, the user must manually access the
source each time they want to check for updates. To address this problem and make it easier for users to receive time sensitive information of interest, such as news and financial data, as soon as such information is available, "push" technology (also known as "agent" or "alert" technology) has been utilized to send preselected topics of information directly to a user's desktop as the information becomes available.

In one type of system, news headlines are provided to the user in the form of scrolled text which is positioned at the bottom of a monitor or computer screen. A program running on the user's machine processes headline information transferred from one or more remote locations and displays it appropriately. This type of display is known as a "ticker". Generally speaking, a ticker is comprised of two elements: text and graphic. Both elements are displayed within the scrolling portion of the ticker. In addition, in a network environment, the text and graphic elements may be associated with a uniform resource locator ("URL") through which the user may access additional remote information related to the specific ticker element over the network. Typically, when a user selects an element in the ticker which is associated with a URL, appropriate network software, such as an Internet browser, is launched and provided with the URL for subsequent processing.

One consideration which must be addressed when using this type of technology, however, is cost since someone must pay to support the generation of timely and appropriate headlines. Because many users of the Internet are reluctant to pay for a subscription type service, it is preferable to generate the necessary revenue through the use of advertising. To this end, graphic advertising, in the form of "banners" or "logos" has been included in the data provided to the user and such advertisements displayed both outside and within the scrolling portion of the ticker.

Unfortunately, conventional Internet ticker technology treats advertising within the scrolling portion as a static element, similar to a basic block of text. Thus, such ads do not necessarily draw the user's attention.

Another drawback with conventional tickers systems is that the produced tickers are horizontal in nature. While a recently developed technique permits a user selectively rotate a ticker and position it vertically along one side of a
computer display, the entire ticker, including headline and advertisement text is rotated as well, making the displayed data difficult to read. A further drawback with conventional ticker programs is such programs do not adequately support multi-linguistic information, including the display of information in more than one language using different alphabets, in possibly different character orientations.

Accordingly, it is an object of the present invention to provide a method and system for displaying animated graphic banners and icons within a scrolling ticker display.

Another object of the invention to provide a vertically scrolling ticker in which the displayed elements are properly oriented to increase readability and also to properly display text information in languages that are traditionally read vertically.

Yet a further object of the invention to provide a scrolling ticker which will properly display foreign characters.

**SUMMARY OF THE INVENTION**

These and other objects are achieved by a system and method for integrating animated graphical elements in a scrolling display window as disclosed herein. A computer system is provided which generates a ticker display element containing a scrolling graphical ticker element. The scrolling portion of the ticker displays both text, such as headlines, and graphics, such as advertisements.

According to one aspect of the invention, elements which are displayed within the scrolling ticker may be animated, even as they are scrolled across the ticker window. The animated frames can be predefined graphical elements, "live" data, such as video images, or images which are dynamically generated as needed. Data for headlines and graphical information are preferably obtained by externally generated "channel" data files which are provided by one or more remote data servers through a network, such as the Internet, although headlines may also be generated locally based on local data sources.

Using animation within the scrolling portion of a ticker, as opposed to conventional static graphic elements or animated elements which are placed
separately from the ticker itself, provides an exciting and attention-getting display which is more effective at drawing a user's attention than conventional systems. In addition, the ticker according to an aspect of the present invention is configured to dock along an edge of the computer display and resize the display so that the ticker does not interfere with other application program displays. Thus, the animated displays are presented while a user is otherwise engaged with various elements on the computer display, further increasing the likelihood that the advertisement will be viewed.

According to another aspect of the invention, a given instance of the ticker may be displayed and scrolled either horizontally or vertically. When in the vertical position, the displayed elements are selectively rotated to appropriately provide vertically or horizontally aligned elements in the vertically scrolling window. When displaying foreign language text, the ticker orientation and scrolling direction may be automatically configured to properly display texts which are readable from right to left, such as Hebrew, and, text which is properly read from top to bottom, such as Chinese.

**BRIEF DESCRIPTION OF THE DRAWINGS:**

The foregoing and other features of the present invention will be more readily apparent from the following detailed description and drawings of illustrative embodiments of the invention in which:

Fig. 1 is a diagram of a network environment for use with the system of the present invention;

Fig. 2 is a representation of the structure of a channel data file;

Figs. 3a-3d are illustration of a preferred configuration of a ticker display according to the invention and its placement on a display device;

Fig. 4 is an illustration of the structure of various data files which are maintained in local storage by the ticker program;
Fig. 5 is an illustration of a ticker window along with an associated ticker buffer and supporting data according to a preferred implementation of the invention;

Fig. 6 is a flowchart summarizing the preferred implementation of the ticker setup and display routine; and

Figs. 7a-7c illustrate various ticker and text orientations.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

Fig. 1 is a diagram of a network environment 10 suitable for the delivery of information and advertisements to a particular user's computer 12. The network may be public or private, wired or wireless, or any combination of LAN and WAN systems. The Internet, a corporate intranet, the public telephone system, or a paging network are examples of suitable networks. The user 12 connects to the network 10 through a conventional data connection, which may be a modem, ethernet cable, wireless link, etc. Although the user device 12 is discussed specifically herein as a "computer", the present invention may be applied to other types of programmable data display systems with pixel oriented screens, such as televisions, handheld "personal assistants", and large screen video "billboard" displays, as well as less conventional devices data display devices, such as pagers, cellular telephones, and other consumer electronic devices.

Also connected to the network 10 are one or more remote data sources 14, 16, 18. Each remote site 14-18 generates data for one or more "channels", each of which contains a series of headlines, preferably grouped by topic, and associated pointers to network locations where further details about each headline are located. Channels are preferably arranged by topic. For example, different channels may be dedicated to general news, stock quotes, weather, company announcements, etc. Preferably, each pointer is a uniform resource locator ("URL") which identifies the network location and the name of a file which contains details and/or the full content related to the associated headline. Fig. 2 is an illustration of a sample channel file 40 containing headlines 42 and associated URLs 43 which identify the location of
additional detail files. It can be appreciated that the data pointed to by a headline URL 43 need not reside on the same storage device as the channel file, and instead may reside anywhere on the network 10. It can also be appreciated that because the URLs in a channel file 40 merely point to separate headline details files, a given URL may identify any type of file which provides additional information related to the headline, such as text, audio, or video, or a combination of the three. Whereas a URL identifies the location and protocol necessary to access a particular piece of data on a computer network, proprietary extensions to the common URL addressing scheme may be necessary to identify data that is not available via a standard URL, as will be appreciated by one of skill in the art. In addition, while channel files are generally provided from remote sources, they may also be generated from locally, from such sources as a database of daily quotations or jokes, for example, or a program which generates headlines warning the user of low disk space or other system parameters.

Each remote site 14-18 may also maintain one or more graphic images, such as advertisements from sponsors of the particular headline server, although this is not required. According to one embodiment of the invention, each advertisement is a graphic image or image sequence identified by a file name or a URL pointer to a file, where the file may comprise a static image, such as a BMP or JPG file, or an animated advertisement, in the form of, for example, a sequence of image frames stored as a GIF or AVI file. Alternatively, the advertisement may be a video stream or even an embedded software routine, such as a Java script, which includes code to dynamically create particular graphical images. A sequence of advertisements can be configured as an advertisement channel file similar to a headline channel file and, as with the headline details files, the image file may physically reside on a different server than contains the advertisement channel and identifier. One or more advertisement channel files may also be maintained on a special advertisement server which is separate from the news-providing remote sites.

At the user's terminal 12, a headline gathering and display program, hereafter referred to as "the ticker program", is run, preferably under a window-based or other graphical operating system, such as Windows NT or Windows 95. The
operation of the ticker program will be discussed in more detail below. In brief, the
ticker program receives headline channel files and advertising channel files, processes
and formats the specified text and graphical data, and displays headlines and
advertisements in a scrolling window. According to a primary aspect of the invention,
the scrolling ticker mechanism is configured to integrate animated graphical elements
and textual headline data in a single integrated ticker window and to further permit the
graphic elements, such as advertisements, to be animated.

Fig. 3a is an illustration of a preferred configuration of a ticker display
20 according to the invention. The ticker 20 is configured as a horizontal toolbar or
screen element which contains a horizontally scrolling ticker window 22. Preferably,
one side of the tool bar 24 displays an operating system application caption which
may be used, for example, to reposition the ticker 20 as desired on the display. The
ticker can be positioned anywhere on the screen and preferably may be resized along
its length and height. In a preferred embodiment, when the ticker display 20 is moved
to one edge of the screen, the ticker program will automatically "dock" the ticker
display 20, resizing it to the full length of that edge and rotating the ticker display 20
as necessary. As part of the docking process, the user's desktop is preferably resized
so that the ticker display 20 does not cover other applications or desktop icons. Figs.
3b-3d show various positions that the ticker 20 may be placed on a computer display

25 In particular, Fig. 3b is an illustration of a ticker display 20 which is docked at the
top of a display 26, Fig. 3c illustrates a ticker display 20 which is "floating" within
display 26 as a separate program window, and Fig. 3d illustrates a ticker display 20
which is docked on the left edge of display 26 and has been rotated 90 degrees.

One or more buttons 28 may also be provided on the ticker 20 to
provide instant access to commonly used functions. Each button may be assigned one
of various behaviors. A buttons may open a dropdown hierarchical menu of
predefined URLs or may launch a URL immediately when the button is clicked to
provide instant access to a preselected web site, or any Internet or intranet resource. A
"Search" button may be provided, selection of which will open a text entry field over
the ticker window and prompt the user for search criteria that will be used to search
the Internet or a database. Likewise, a "Go to" button may be provided, selection of which prompts the user in the same fashion to go to a specific Internet address, provides a drop-down menu hierarchy of bookmarks imported from an Internet web browser, etc. A main menu button 30 may also be provided. The main menu button 30 provides user access to a variety of configurable preferences and functions such as exiting the program, pausing the ticker, enabling or disabling auto hide, changing the appearance of the ticker, and requesting help.

According to a particular aspect of the invention, a ticker window 22 (Fig. 3a) is provided within the ticker display 20 which is scrollable along a predefined axis of the ticker 20, preferably along the long axis as shown in Fig. 3a, and contains text elements 21, such as headlines, and graphic elements 23, such as animated advertisements. Preferably, each headline is preceded by an icon 25 which identifies the starting point of a headline and also indicates whether the associated content is text, audio, or video. Other content types may also be indicated, such as email, USENET News, Lotus Notes, and Microsoft Exchange. The headline status, such as "read" vs. "unread" or "new" vs "old", may additionally be indicated via the associated icon. The icon 25 may be any supported ticker data type, but is preferably a static or animated graphic. Also, in the preferred embodiment, headlines are displayed in conjunction with the name of the channel from which they were supplied (not shown).

Text elements 21 are generally comprised of alphanumeric characters (letters, numbers, and punctuation) which are combined to form headlines, announcements, and other human readable messages. The text elements are systematically drawn within the ticker window 22 based on raw text supplied to the ticker program. Even if a text element contains formatting information (i.e. bold, italic, underline, color, font name, font size, and position), it is still considered a text
element. However, when text characters are rendered dynamically using processes similar to that described below for animated graphical elements, the characters are more properly considered to be animated elements.

In a preferred embodiment of the invention, text information for headlines and the like is represented using UNICODE encoding, as opposed to ASCII character codes. Unlike ASCII, which provides very limited or no support for foreign languages which do not use the English alphabet, UNICODE, officially called the Unicode Worldwide Character Standard, provides support for the interchange, processing, and display of the written texts of the diverse languages of the modern world. Currently, the Unicode standard contains 34,168 distinct coded characters derived from 24 supported language scripts. Using the UNICODE specification, headlines may be displayed in the ticker using diverse languages such as Japanese, Chinese, and Hebrew, individually or in combination.

A graphic element is generally a picture or image. Although graphic elements may contain representations of text, they are not limited to purely text character. Typically, a graphic element is supplied to the ticker program in the form of a vector or bitmap image. The image file format may be GIF, JPEG, BMP, AVI, MPEG or any other format used for exchanging images. Animated graphic elements are displayed in the scrolling ticker window 22 as a sequence of images, rather than as a single static image. The sequence of images may be displayed slowly, giving a user time to read a message on each frame. Or they may be displayed in rapid succession, such as with a video clip, or an animated advertisement banner, to give the illusion of motion.

Another type of graphical element which may be displayed in the ticker window is a "programmatic graphic element." This element is similar to a URL
linked headline or graphic, except that it includes computer programs or pointers to a computer program or script, such as Java and ActiveX. The identified code is used to dynamically generate the graphically displayed frames on an as-needed basis and can also support other display formats be rendered graphically, such as HTML, XML, VRML, Microsoft Power Point, and MacroMedia ShockWave and Flash.

Advantageously, the inclusion of animation in the scrolling ticker window 22 adds an exciting and attention-getting advertisement medium which is much more effective at catching the user's attention than conventional systems which only display static images. Because an advertisement is only effective if it is noticed and remembered, integrating animation into the moving ticker increases the likelihood that an advertisement will be effective. Preferably, the effectiveness of animated ad banners is further enhanced by directing particular ads to specific users according to each user's interests as indicated by, e.g., the headline channels they are subscribed to, the types or subject matter of headlines the user selects to view in more detail, and possibly other types of demographic information. For example, financial oriented ads may be directed primarily to individuals who subscribe to numerous financial channels or who click on a large number of finance-related headlines. By targeting advertisements to particular users in this manner, the effectiveness of the delivered advertisements is enhanced.

In the preferred implementation of the ticker program, discussed in more detail below, the ticker window 22 displays a portion of an off-screen buffer which is the same height as the ticker window 22 but substantially longer. The buffer is loaded with the data to be shown in the ticker window. By repeatedly changing the window offset relative to the start of the buffer, the ticker can be made to scroll in either direction along the selected axis.
The ticker program is configured with the locations of one or more channels, which may be located on any data server that is accessible to the user via a computer network. On a regular basis, the program accesses the network and retrieves an updated channel file for each active channel currently subscribed to. The updated channel file may replace a currently existing channel file on the local system, augment it, or the contents of an old and new files may otherwise be merged. In the preferred embodiment, channel headlines for a given channel are automatically deleted in accordance with corresponding user defined thresholds that indicate the maximum number of headlines which may be loaded into a particular channel file at one time. A maximum duration a headline may reside in the headline channel file may also be defined wherein headlines which have been present in the file longer than the maximum duration are deleted.

To conserve bandwidth, a pause typically of 5 to 30 seconds preferably takes place between updates of different channels. If 60 channels are in need of an update, the updating of all 60 channels will take place over the duration of somewhere between 5 and 30 minutes depending on the duration. The exact duration of the pause may be predefined or adjusted by one or both of the user and a third party, such as a network manager or channel publisher. Once a channel has been updated, another attempt to update the channel will not be made until the specified duration has transpired. For example, some channels may be configured with an update duration of 60 minutes while others have an update duration of 24 hours.

In addition to downloading headline channel data 40, the ticker program is also configured to retrieve data from one or more advertisement (ad) channels 44. Ad channels 44 are similar to headline channels, except that the content of headline channel files is typically text based data, whereas the content of ad banner
channels is generally comprised of URL pointers (or identifiers) which identify specific graphical data files that will be rendered within the scrolling portion of the ticker (in place of a textual headline). Unlike headline data, where the headline is a simple and short block of text, advertisements are generally graphical elements in one form or another. The content associated with each URL in the ad channel 44 is downloaded in conjunction with the data in the ad channel file itself and stored locally so that it can be displayed without further network access. Downloading can be done as each new pointer is received. Preferably, the ad channel file 44 is examined on a periodic basis to locate any entries which point to data that has not yet been downloaded. When the first such entry is identified, the download process is initiated. Because graphical data associated with an ad entry in the ad channel file may not display properly until it is fully downloaded, preferably, a placeholder, such as such as a rolling animated "polar-rose", is displayed in the place of those ads graphics which are scheduled for display but which have not yet been fully downloaded.

In the preferred embodiment, displayed headlines are grouped according to their source or topical category. For example, in a system subscribed to a Technology and a Business channel provided by CNET, and a sports channel provided by MSNBC, all "CNET / Technology" headlines are first displayed, followed by the "CNET / Business headlines", the "MSNBC / Sports" headlines, and so on. All subscribed channels that have headlines will appear in the ticker eventually. Alternatively, the headlines can be directly accessed via a hierarchical selection menu which is activated, e.g., by right-clicking with the mouse anywhere on the ticker display. The top level of the hierarchy consists of the channel sources (e.g., CNET, MSNBC, etc.). Under each top level menu option appear the specific channels associated with that source (technology, business headlines, etc.). Clicking a
channel name on this menu advances the ticker directly to that channel. As there may hundreds of headlines waiting to scroll through the ticker, this is an effective means for navigating. Unsubscribed channels, and subscribed channels that have no headlines, will not appear in this channel menu selection.

The operating process for the ticker program, with specific regard to the display of animated graphic elements in a scrolling ticker, will now be described with reference to the remaining figures. Fig. 4 is an illustration of various data files (or portions of data files) which are maintained in local storage (disk and/or RAM) by the ticker program. As discussed above, a user subscribes to one or more news channels and the program fetches according to predefined criteria channel files from remote sites. Each channel file contains one or more text-based headlines. Each headline may include a URL pointing to the location of the detailed news file associated with the particular headline. In operation, a number of such headlines are loaded into memory while others are stored on disk. Unlike ad channels, the URL pointers in a headline channel typically point to more detailed information, and not graphical content which is to be displayed in place of textual headlines within the ticker.

In a further enhanced embodiment of the invention, however, the pointer associated with a headline can point to a specific graphical element which is to be displayed in place of the text data. For example, if the headline is a weather update, a corresponding graphical element may be an icon illustrating that the weather will be sunny, rainy, etc. When a channel file contains both a textual headline and a URL pointer to a such a graphical element, the graphical element takes precedence and will be displayed in the ticker in place of the provided headline text in a manner similar to the display of graphical advertisement data. If the
graphical data associated with the headline is unavailable or has not been fully downloaded, the text headline will be displayed instead. Once the graphic is downloaded, the text headline is preferably utilized in the form of a "tool tip", a tiny popup window that appears automatically when the mouse pointer enters the region of the computer display currently occupied by the associated graphic or text element.

In addition to being subscribed to one or more headline channels, the ticker program is subscribed to one or more ad channels 44 which each contain a sequential listing of advertisement sources and pointers. In a preferred file structure, shown in Fig. 4, the ad channel file 44 comprises a sequence of URL entries, each ad having a banner URL entry 46 and an optional click URL entry 47. The banner URL identifies a source for the advertisement to be displayed. Suitable sources include an image file 48, an animated "movie" image 49, a video source 50, such as a streaming video port, and a local or remote software routine 51 which generates graphical images. The click URL 47 associated with a particular banner URL 46 indicates the action to be taken when a user selects, e.g., by clicking, the associated banner graphic as it is displayed on the ticker. The click URL 47 can identify the location of a file 54 to be accessed, such as an HTML web page, an image or video 52 to display, a program 53 to execute, etc. Although the headline channel file 40 and the ad channel file 44 are shown as separate data files, those of skill in the art will recognize that the data elements can, in fact, be stored in a single data file.

Preferably, only the ad graphics currently displayed in the ticker, or about to be displayed, are cached in RAM. The remaining ad graphics, once they have been fetched or rendered, are stored on a local hard disk or other local storage. In the preferred embodiment, when an image is loaded into memory, it is converted from its native file format (i.e., GIF, JPEG, AVI, MPEG, etc.) to a homogenized
internal format, such as a bitmap image. If the native file format contains an
animation or video sequence, then the consecutive frames are preferably appended to
each other to create a single static bitmap image 55 which contains each image frame
56 and a header 57 which indicates the number of frames and, if necessary, the size of
each frame. (See Fig. 4.) For example, a 100 x 38 pixel 3 frame ad banner in GIF
format would be converted to a 100 x 114 pixel static bitmap image. As part of the
conversion process, the color palette may be adjusted to match the color palette
capabilities of the host computer. Because the number of frames in the animation and
the size of each frame is known, specific frames may be extracted from the static
bitmap image 55 as needed. Those versed in the art will recognize that there other
methods to store a sequence of images in memory such that frames in the sequence
can be quickly accessed as needed so that the frames can be efficiently rendered as an
animated sequence. For example, an array linked or linked list of images could be
used. Alternatively, each frame could be dynamically extracted from the original
format as needed.

As shown in Fig. 5, the ticker routine maintains an off-screen buffer 60
which contains the data which is displayed in the scrolling portion 22 of the ticker.
The buffer 60 is at least the same height in pixels as the scrolling ticker window 22
but is significantly wider, preferably at least two times the width of the scrolling ticker
window 22. For a vertically oriented ticker, discussed in more detail below, the
buffer 60 is at least the same width as the vertical scrolling portion 22 of the ticker,
but preferably at least twice as high.

As illustrated in Fig. 5, the data in the buffer 60 can come from a
variety of sources. Headline data 42 is supplied by one or more headline channel files
40. Headlines are preferably selected from each headline file 40 in sequence and a
pointer 68 may be provided to indicate the next headline to include. Animated data is
supplied to the buffer 60 in accordance with its source. If an animation has been
reformatted into a static bitmap image 55, individual frames 56 can be copied into an
appropriate portion of the buffer 60 at the selected frame rate. In Fig. 5, pointer 66 is
used to indicate the frame with the static bitmap to copy. A separate procedure 62
may also be used to fetch each ad frame as it is needed. The fetch procedure 62 may
retrieve frames from remote locations, generate them dynamically, or retrieve the
frames from a predefined frame data array, such as a static bitmap image 55.

At any given point in time, a view into a selected portion of the buffer
is provided by copying data from a "windowed" area 61 of the buffer 60 into the
ticker window 22. The windowed area 61 is defined by an offset point 64 and an end
point 65, which is the offset pointer 64 plus the width (or height) of the scrolling
portion 22 of the ticker. The ticker window 22 is made to scroll by periodically
incrementing or decrementing the offset 64 of the view into the buffer 60. An
increment or decrement of zero will appear to pause the ticker. However, because the
windowed portion 61 of the buffer 60 is still copied to the scrolling portion 22 of the
ticker, changes made to data in the off-screen buffer 60 will still be reflected on-
screen. This allows animated graphic elements to continue to be animated, even when
the ticker is paused. Advantageously, because the buffer 60 is static (i.e., it does not
scroll), the position for each data item is placed in the buffer remains constant. This
simplifies the inclusion of animated elements since copying of data into the buffer 60
can be done without regard for the speed or direction of the scrolling ticker window
22.

Although other display techniques may be used, as will be appreciated
by those skilled in the art, an off-screen buffer 60 as described herein is preferable
because it provides for better visual cosmetics and better performance. Without an off-screen buffer, it would be necessary to write changes directly to the display which would cause severe flickering. Better performance occurs since the effort of rendering the text and ads into their appropriate places is typically only done once every several hundred screen updates. Without the off-screen buffer it would be necessary to perform these calculations with every screen update, which happens several times a second. Advantageously, with buffering, CPU intensive calculations occur only once every five to twenty seconds.

Those skilled in the art will also recognize another alternative to the off-screen buffer solution. Modern video cards frequently support an operation that moves a rectangular region of pixels from one portion of the screen to another portion. Because this operation is implemented by the video card, and not the ticker program, it is extremely fast and efficient. However, this method is only applicable if there are no animated or dynamically generated text or graphical elements in the scrolling ticker.

In the preferred embodiment, each data element, i.e., a headline, advertisement graphic, etc., which is to be displayed is stored in memory as a discrete Ticker Element Object using oriented program techniques. Each ticker element object is embedded with the necessary information to properly generate appropriate data to be placed into the buffer. Specific object types may include, for example, Headline Ticker Element Objects, Stock Quote Ticker Element Objects, and Animated Graphical Ticker Element Object. There may also be specific types of ticker elements for Java, ActiveX, and HTML. Each object type may have any number of instances.

The specific data element objects are a superset of a Generic Ticker Element Object, and each specific object inherits the definition (data elements and
procedure calls) of the Generic Ticker Element Object. The common information may be a URL to invoke when the object is clicked, the width of the object when it is rendered graphically, a procedure or method for initializing the object, a procedure or method for rendering the object graphically, a procedure or method for a conditional rendering of the object, and a procedure or method for destroying the object. These procedures (a.k.a. "methods" in C++) are "pure virtual". That is, they are defined in the Generic Object, but implemented in the descendent objects. Therefore all descendants of the Generic Ticker Element Object know how to initialize and render themselves graphically, but different objects may render themselves in different manners.

In addition, all Ticker Element Objects know how to initialize themselves, given the appropriate parameters, and how to destroy themselves, when no longer required. For a Headline Object, parameters may include the actual text of the headline and the URL to go to when the headline is selected. For an Ad Banner graphic Object the parameters may include the URL and the name of the file containing the ad banner. The specific method of initialization is largely dependent on the type of data being presented. Different methods will be apparent to those of skill in the art.

In this embodiment, all Ticker Element Objects also know how to render themselves graphically onto an off-screen (in-memory) buffer or an on-screen computer graphic display. For example, the ticker process may ask an object to draw itself into an off-screen buffer at a specified x-y coordinate. If it is a Headline Object, it will select the appropriate font, determine whether or not the font the needs to be rotated 90 degrees for a vertical scrolling ticker, and utilize Windows GUI system calls to render the headline text directly on to the display surface provided by the
ticker routine. Likewise an Ad Banner Object would utilize specific routines which draw the ad banner into provided display surface. An ActiveX, COM, Java, or HTML Ticker Element Object may function in a similar fashion.

Every time the on-screen ticker is to be updated from the buffer, the process (described below) checks to see if the buffer itself needs updating. This may involve asking each Ticker Element Object to update it’s corresponding graphical representation in it’s respective position in the off-screen buffer. Although headlines could change as the ticker scrolls by, generally they are static elements and do not need to refresh their graphical representation in the buffer every time the buffer is copied onto the screen. Preferably, for efficiency, Headline Ticker Element Objects are only rendered into the buffer once each time they scroll across the screen.

Ticker Element Objects can be called with a request to draw themselves conditionally, that is, to draw themselves only if they have changed since the last time they were drawn. For example, an Ad Banner Object may be asked ten times a second to paint an appropriate graphic into the ticker, but only if changes have occurred since the last time. It may be the case that this instance of the Ad Banner Object contains a sequence of images that are displayed once a second. Therefore, 9 out 10 times, the Ad Banner Object will decline to paint itself in this scenario.

Graphic banners are preferably updated once a second by default, or at the interval specified by the creator of the artwork. Since the ticker itself is updated from the buffer several times a second, the Ad Banner Ticker Element Object may often decline to refresh it’s graphical representation in the ticker. In contrast to a graphic ad banner, a Video Ticker Element Object may have a new video frame every time the ticker process asks it to update it’s graphical representation in the ticker. In this case, with every increment/decrement of the position of the ticker a frame of
video will be copied into the buffer. Advantageously, because each Ticker Element Object takes the responsibility of determining when in it appropriate to update the buffer, the main display program is freed from having to process this information separately. Preferably, for optimization purposes, only those Ticker Element Objects which are actually visible on the computer graphic display are asked to update their graphical representation in the buffer. For example, when Video and Ad Banner objects scroll out of view, they no longer need to be refreshed into the buffer.

Fig. 6, is a flowchart summarizing a preferred method for ticker setup and display. In the preferred embodiment, the ticker program which performs this method is executed as a program thread of execution in a Microsoft Windows operating system and speaks directly to Winsock, a TCP/IP networking library standardized across Windows 3.1, 95, 98, NT, 2000, and CE. All communications initiated by the ticker program are via the Hypertext Transfer Protocol (HTTP). Such HTTP transactions are composed of standard HTTP commands, such as "GET" and "POST" requests. The program also supports these communications through corporate firewalls via standard HTTP Proxy servers. Those versed in the art will recognize that other communications protocols and firewall solutions may be used as a substitute or as a compliment to HTTP.

Turning to Fig. 6, when the ticker is first initialized, a number of Ticker Element objects, i.e., text or graphic items which are to be displayed on the ticker, are requested (step 100) from a data access routine that serves as an interface between the ticker display routine and the various headline and ad channel files. In one embodiment, each time the interface routine is called, it returns data related to a particular headline or ad from an appropriate channel files according to a predefined selection sequence. The data associated with the returned ticker element objects (i.e.,
a headline text or ad graphic) is loaded into memory (step 102). The returned ticker
element objects are preferably stored in the form of a linked list. Enough objects are
requested to provide sufficient data to fill the buffer 60. Accordingly, the total
number of elements which must be requested is dependent on the amount of buffer
space each element requires when rendered or printed and the size of the buffer.
Preferably, the sizes of each element are predetermined. However, they may be
calculated on-the-fly.

The ticker window offset is then reset (step 104) and the data for each
element in the list of ticker elements is processed as required and appropriate to
generate display data, which data is then mapped in the proper order into the data
buffer 60. (Step 106). At this point, the data buffer 60 contains the graphical data
which will be displayed in the ticker window 22, as graphically illustrated in Fig. 5.

Once the data is loaded in the buffer 60, the routine enters the main
scrolling window loop. Starting at the window offset point 64, the buffered data is
copied into the visible ticker window 22 on the video display (step 108). Next, each
animated ticker element in the buffer is examined to determine whether the graphical
element needs updating (step 110). If an update is required, the next graphic frame in
the sequence is retrieved and loaded into the buffer (step 114). An update may be
necessary, for example, to load the next frame of a predefined bitmap array 54 (see
Fig. 5), to fetch the most current frame of a live video sequence, to generate a
dynamically created animation according to a particular software routine, etc. After
all necessary buffer updates are performed, the window offset 64 is incremented or
decremented, according to the defined ticker scrolling direction and speed (step 116),
which speed is typically 0 to ±30 pixels. The greater the increment or decrement, the
faster the ticker scrolls. An increment/decrement of zero will pause the ticker, but
animated elements will continue to be animated.

It can be appreciated that at some point, the leading or trailing end of the ticker window will move beyond the edge of the buffer. When this condition is detected (step 118), additional ticker elements are requested from the data access routine to provide more data to display (step 120). The new elements are added to the end of the Ticker Element list that corresponds to the end of the buffer which was overrun by the ticker window (step 122). Ticker Element Objects are then removed from the opposite end of the list until the data from the list elements will fit within the buffer (step 124).

The program then loops back to step 104 to reset the ticker window offset (this time ensuring that the new offset is appropriately aligned with the old buffer data and data currently displayed in the buffer). The buffer data is then mapped, updated, and the window offset adjusted, as described above. Although preferably the display process is implemented as a separate thread of execution, a substantially similar process can be implemented as an asynchronous state machine driven by regular clock ticks.

Preferably, all multi-frame animated banners in the off-screen buffer are updated with the next appropriate frame at an interval defined by the creator of the animated banner, i.e., once each second. The appropriate frame to use may be determined by maintaining a "frame-index" pointer 66 for each ad banner 55, as shown in Fig. 5. Once the last frame for a given ad banner is displayed, the frame-index pointer 56 is reset to the first frame. The new ad banner frame will appear the next time the corresponding portion of the buffer 60 is copied into the visible ticker window 22. As is common practice, the creator of the artwork may use a different frame reach for each and every frame in an animated banner.
The process for real-time streaming video is similar. However, the loading, converting, and caching an image file in memory is not required in this case. In this case, when a new graphic is needed, the last real-time image obtained by an appropriate video streaming program, such as Real Video, is retrieved, converted to a homogenized bitmap format, and used to update the off-screen buffer.

In yet another embodiment, a software routine 62 is called by the main buffer update routine with a request for the next frame image for a particular ad graphic. The image is returned in a homogenized bitmap format and used to update the appropriate portion of the buffer 60. The software routine analyzes the type of ad graphic displayed and automatically makes the appropriate image selection. For example, the routine may return the next bitmap from an image array 54, extract the most current image from video buffer, or call specifically identified programs, such as Java scripts, which generate the requested graphic image on the fly.

As mentioned above, to keep track of what images are displayed in the ticker and where, the ticker program maintains a list of the Ticker Elements Objects currently in the buffer. Furthermore, a subset of this list indicates which of the elements in the buffer are currently being rendered to the screen. This optimization means that only those animated or dynamically generated elements that are currently in the visible portion of the ticker need be updated, saving CPU time. Preferably, a linked list is used, although other data structures such as an array would be adequate. Each object in this list is a particular type of Ticker Element Object. Because each object, when graphically rendered into the buffer, has a certain definite width in pixels and the buffer has a definite width, the list need only have enough elements to fill the buffer. For example, if 5 headlines, 1 ad graphic, and another 3 headlines are positioned in the buffer, filling it from end to end, then there need only be 5 Headline
Ticker Element Objects, followed by 1 Ad Banner Ticker Element Object, followed by 3 Headline Ticker Element Objects in memory.

Each element in the list is associated with a particular position in the buffer, i.e., a starting pixel and a length. In addition, each element displayed in the ticker window 22 may have an associated URL. Selecting an element (i.e., by left-clicking with a mouse), will launch an appropriate web data access program, such as a web-browser, and pass it the associated URL for processing. Clicking and dragging in the ticker window 22 (without releasing the mouse button) allows the user to advance the ticker in either direction, without opening a headline. After a click and drag, the ticker will continue to scroll in the direction in which it was advanced. The speed at which a user drags the ticker may be used to adjust both the speed and the direction of the scrolling.

More specifically, when the user clicks on an area in the ticker 22, the mouse coordinate is translated into a coordinate relative to the top left corner of the on-screen portion of the ticker. The offset is added to the X component of the coordinate to compute the corresponding X-Y coordinate in the off-screen buffer. It is then determined which Ticker Element Object was rendered into the off-screen buffer at that position. This can be computed by looking for the first object in the list that has a left edge to the left of the X coordinate and a right edge to the right of the mouse coordinate. Once this is done, the corresponding object is asked to process the mouse click. Typically this involves calling a routine that accepts the associated URL and opens a web browser at that address, or some other program, such as a streaming Video display program, that could process the URL. However, any other action may be associated with a Ticker Element Object. For example, a programmed element
could show a rendered video clip in its portion of the ticker in response to a user clicking on it.

When the ticker is created, for example, at start up or when the ticker is switched to vertical mode and reset, or reset for other reasons, a sufficient number of Ticker Element Objects to fill the off-screen buffer are instantiated and copied into memory. As discussed with respect to steps 102 and 120, above, during the ticker initialization and update process, Ticker Element Objects are also requested. Preferably, elements are selected by a selection procedure which will return a Ticker Element Object instantiated in memory using methods known to those skilled in the art. The ticker process may or may not specify the type of object.

The selection of the various elements, such as headlines and ads, for display in the buffer may be made in a variety of ways. Preferably, an index into a list or database of headlines is maintained. Each time a headline is requested, the next headline in the index is selected. The selected headline is then instantiated to create a Headline Ticker Element Object, initialized with the headline text, URL, and associated information, and then returned to the main ticker routine.

After a predetermined number of text elements, such as headlines, have been returned, the selection procedure can return a graphic element, such as an ad banner. A separate index into a list or database of ad banners may be maintained for this purpose. The routine accesses the next ad banner in the list, as indicated by, e.g., an index, and loads the associated data types into memory (ad banner URL, the name of the file containing the graphical representation of the ad banner). Alternatively, selection of the advertisements to display (and possibly the advertisements to download) are made in accordance with predefined selection criteria, such as the headline files the particular user is subscribed to. Once an ad is selected for display,
the routine instantiates an instance of the Ad Ticker Element Object and initializes it with the provided data. The Ad Ticker Element Object then loads the associated graphical file into memory and converts the original graphical representation into a format compatible with the Windows operating system GUI routines, if necessary.

Alternatively, the object may defer processing associated data until receiving a request to draw itself into the buffer.

Likewise, the selection routine may determine, randomly or based on predetermined conditions, that a Video Ticker Element Object should be added to the ticker. Such an object would then be instantiated in memory and initialized with information such as a URL the Internet address of the Video server. The Video Ticker Element Object, as it is initialized, may automatically establish a connection to the video server so that video frames are available as needed.

The discussion above has focused on a horizontally oriented ticker 22. According to a further aspect of the invention, and as shown in Fig. 3d, the ticker may be positioned vertically. To produce a vertically scrolling ticker window, a vertical buffer is generated and filled with data in a manner analogous to the setup of a horizontal buffer.

In a first embodiment of this aspect of the invention, Ticker Element Objects are configured to automatically rotate their data by 90 degrees before it is placed into the vertical buffer. Alternatively, the buffer data may be rotated as it is copied into the scrolling ticker window 22. As a result the entire ticker is viewed as having been rotated by 90 degrees. For example, Fig. 7a illustrates a horizontal ticker window 22 with a headline object 70 and a graphical ad element 72. Rotating the ticker in the manner discussed above results in the ticker as illustrated in Fig. 7B,
wherein rotated ticker 22' includes a rotated headline object 70' and rotated graphical ad element 72'.

According to a second embodiment of this aspect of the invention, individual text and graphic elements are selectively broken into their constituent parts. Each part is then rendered horizontally and the horizontally rendered parts are vertically arranged to produce a vertical display having horizontally aligned elements. For example, as shown in Fig. 7a, a headline element 70 is composed of a number of individual characters 74. When a headline is displayed vertically, it is broken up into component parts comprising one or more characters, such as a word or even an entire line. Each character is rendered in a normal orientation as it is placed into the buffer 60. Preferably, individual graphical elements 72 are also shown in their normal (horizontal) orientation. The resulting vertical display 22", shown in Fig. 7c, includes a vertical headline 70" having horizontally readable characters and a horizontally aligned graphic 72.

Use of a vertically oriented ticker as shown in Fig. 7c has several advantages over conventional horizontal tickers. For example, it permits the proper display of textual information in languages which are read vertically (top to bottom, or bottom to top). In addition, small horizontally readable textual elements, such as stock quotes, may be rendered as discrete graphical elements (which are not rotated), as opposed to text elements, which are broken up into stacked characters, to provide a vertically scrolling stock ticker. It can be appreciated that stock quotes formatted in this manner may be treated similarly to other images which are dynamically generated as needed.

In a further aspect of the invention, the language being displayed is determined, for example, by analyzing the UNICODE characters provided in supplied
headline text, or by a language identifier include in a channel file. Each language type may be programmed with a default ticker orientation and direction to properly display multi-language data. For example, the default for English text may be a horizontal ticker scrolling from right to left (so that the text may be read from left to right as it goes by). For Hebrew text, the default may be a horizontal ticker which scrolls from left to right, so that the text may be properly read from right to left. Similarly, if Chinese characters are detected, the ticker may default to a vertical orientation, scrolling from bottom to top, while displaying each character in a normal horizontal orientation. Advantageously, providing support for UNICODE character sets permits a user to subscribe to data channels which may provide headline data in a wide variety of languages with the knowledge that foreign (i.e., non standard English) characters will be displayed correctly.

In the preferred embodiment of the invention, the user is also able to customize both the look and feel of the ticker as shown in the screen to best fit into the user's overall display environment as well as their own personal preferences. For example, a user may select between Standard, Modern, and Stealth "modes", each of which has a pre-defined color scheme.

Although use of only a single ticker has been discussed, according to the invention, multiple tickers, perhaps in different orientations, may be provided on a given display in order to show different information. In one particular configuration, a horizontal ticker is positioned at the top of the screen to display headline information. A second, vertical ticker is positioned to one side of the screen and used to display small data blocks, like graphically rendered stock quotes, as discussed above. As with a single ticker, a user may position the several tickers on the display as desired. Information may be selectively directed to particular tickers by
subscribing each ticker to different channels, for example one ticker may be
subscribed only to a stock quote channel, while the other ticker is subscribed only to
headline channels.

While the invention has been particularly shown and described with
reference to preferred embodiments thereof, it will be understood by those skilled in
the art that various changes in form and details may be made therein without
departing from the spirit and scope of the invention. For example, while the invention
has been discussed primarily with regard to headline tickers containing advertising
graphics, the invention can be applied to a variety of other scrolling displays,
preferably windowed, containing mixtures of text and graphic elements.
CLAIMS:

1. A method for displaying animated graphical elements comprising a plurality of frames in a scrolling ticker window on a computer display, the ticker window being scrollable along a length, the method comprising the steps of:

   1. allocating a buffer with a size larger than the ticker size;
   2. retrieving a plurality of ticker element objects for display in said ticker, each ticker element object representing one of a static text element, a static graphical element, and an animated graphical element;
   3. mapping at least some of said retrieved ticker element objects to said buffer;
   4. repeatedly copying a portion of said buffer beginning at an offset point into said ticker window and repositioning said offset point in accordance with a defined ticker scroll speed and direction to thereby produce an autonomously scrolling display in said ticker window;
   5. concurrently with said copying and repositioning step:
      1. identifying any animated graphical element mapped to said buffer which requires updating; and
      2. for each identified animated graphical element, retrieving a respective next frame and refreshing a corresponding present frame in the buffer with a next frame.

2. The method of claim 1, further comprising the steps of:

   1. detecting a condition when the size of the ticker window exceeds that between the offset point and an end of the buffer;
   2. retrieving additional ticker element objects for display in said ticker
when said detected condition is true; and

mapping said retrieved additional ticker element objects to said end of
the buffer.

3. The method of claim 1, wherein:

a particular ticker element object comprises an animated graphical
element including a plurality of predefined frames in a native file format;
the step of retrieving the particular ticker element object comprises:
retrieving the predefined frames associated with animated graphical
element;
converting each of said retrieved frames to a bitmap format; and
generating a static bitmap image comprising said retrieved converted
frames and a size data field indicating the number of frames in said static bitmap
image;
the step of mapping comprises mapping a first frame from said
respective animated graphical element to said buffer;
the step of retrieving a respective next frame for the particular ticker
element object comprises extracting a next frame from the respective static bitmap
image.

4. The method of claim 1, wherein:
a particular ticker element object comprises an animated graphical
element including a video data stream;
the step of retrieving the particular ticker element object comprises
retrieving a first video frame from said video data stream; and
the step of retrieving a respective next frame for the particular ticker
element object comprises retrieving a subsequent video frame from said video data
stream.

5. The method of claim 1, wherein:
a particular ticker element object comprises a programmatic graphic
element;
the step of retrieving the particular ticker element comprises retrieving
computer code for generating a plurality of graphical images when executed;
the step of mapping a first frame from the particular ticker said
respective animated graphical element to said buffer comprising the step of executing
the retrieved computer code to generate the first frame; and
the step of retrieving the respective next frame for the particular ticker
element object comprises the step of executing the retrieved computer code to
generate the next frame.

6. The method of claim 1, wherein said ticker element objects comprise
data retrieved from a remote location.

7. The method of claim 1, wherein:
each ticker element object is implemented in an object oriented
programming environment and is embedded with information necessary to
autonomously generate appropriate data to be mapped to said buffer;
the step of mapping comprises the step of instructing each respective
ticker element object to render itself within an indicated portion of the buffer.
8. A method of generating a scrolling ticker on a computer display, the
ticker comprising a ticker window being scrollable along a length, the method
comprising the steps of:

- retrieving a plurality of headline entries from at least one selected data
  channel located on a remote site, each headline entry comprising a text headline for
display in said ticker;

- retrieving a plurality of non-headline entries from a remote source,
each non-headline entry comprising a banner pointer indicating a source of at least
one frame of graphical data for display in said ticker;

- storing said retrieved headline and non-headline entries in at least one
  local channel file;

- establishing a ticker window on said display,

- allocating a buffer with a length greater than the length of the ticker;

- selecting at least one of a headline entry from said headline channel
  files and a non-headline entry from said ad channel file for display in said ticker;

- converting the respective text headline for each selected headline entry
  into graphical data and storing said graphical data in said buffer;

- retrieving for each selected non-headline entry at least a first frame of
  the graphical data identified by the associated banner pointer;

- mapping the first frame from each selected non-headline entry to said
  buffer;

- repeatedly copying a portion of said buffer beginning at an offset point
  into said ticker window and repositioning said offset point in accordance with a
defined ticker scroll speed and direction to thereby produce an autonomously scrolling
display in said ticker window;
concurrently with said copying and repositioning step:

identifying selected non-headline entries associated with a plurality of frames of graphical data and which require a frame update;

for each identified non-headline entry, refreshing a corresponding present frame in the buffer with a next frame.

9. The method of claim 8, wherein:

each headline entry further comprises a details pointer to a data file associated with the corresponding text headline;

each non-headline entry further comprises a click pointer to a data file associated with the corresponding graphical data;

the method further comprising the steps of:

detecting a selection by a user of a region in the ticker window;

identifying an entry in said buffer corresponding to the selected region;

retrieving one of a details pointer and a click pointer associated with the identified entry; and

processing the data file indicated by the retrieved pointer.

10. The method of claim 8, wherein the text headlines for each headline entry are represented using UNICODE encoding.

11. The method of claim 8, further comprising the steps of:

detecting a condition when the size of the ticker window exceeds that between the offset point and an end of the buffer;

selecting at least one of an additional headline entry from said headline
channel files and an additional non-headline entry from said ad channel file for
display in said ticker when said detected condition is true;

converting the respective text headline for each selected additional
headline entry into graphical data and storing said graphical data in said buffer;

retrieving for each selected additional non-headline entry at least a first
frame of the graphical data identified by the associated banner pointer; and

mapping the first frame from each selected additional non-headline
entry to said buffer.

12. The method of claim 8, further comprising the steps of:

retrieving updated headline entries from said selected data channels on
a periodic basis; and

storing said retrieved updated headline entries said corresponding
headline channel files.

13. The method of claim 8, wherein the step of retrieving updated headline

entries comprises:

retrieving updated headline entries for a first particular selected data
channel; and

pausing a predetermined period of time prior to retrieving updated
headline entries for a second particular selected data channel.

14. A method for displaying animated graphical elements comprising a
plurality of frames in a scrolling ticker window a computer display, the ticker window
being scrollable along a length, the method comprising the steps of:
allocating a buffer with a size larger than the ticker size;
retrieving a plurality of ticker element objects for display in said ticker;
determining the orientation of said ticker;
when said ticker orientation is horizontal, mapping said retrieved ticker
element objects to said buffer in a horizontal orientation;
when said ticker orientation is vertical:
separating said retrieved ticker element objects into discrete
components; and
mapping said retrieved ticker element objects to said buffer in a
vertical configuration, wherein each discrete component is mapped in a horizontal
orientation; and
repeatedly copying a portion of said buffer beginning at an offset point
along said length into said ticker window and repositioning said offset point in
accordance with a defined ticker scroll speed and direction to thereby produce an
autonomously scrolling display in said ticker window.

The method of claim 14, wherein:
each ticker element object represents one of a static text element, a
static graphical element, and an animated graphical element;
the method further comprising the steps of, concurrently with said
copying and repositioning step:
identifying any animated graphical element mapped to said
buffer which requires updating; and
for each identified animated graphical element, retrieving a
respective next frame and refreshing a corresponding present frame in the
buffer with a next frame.

16. A computer program for displaying animated graphical elements
comprising a plurality of frames in a scrolling ticker window a computer display, the
ticker window being scrollable along a length, the computer program comprising:
a buffer program module allocating a buffer with a size larger than the
ticker size;
a data retrieval program module retrieving a plurality of ticker element
objects for display in said ticker, each ticker element object representing one of a
static text element, a static graphical element, and an animated graphical element;
a mapping program module mapping at least some of said retrieved
ticker element objects to said buffer;
a display program module repeatedly copying a portion of said buffer
beginning at an offset point into said ticker window and repositioning said offset point
in accordance with a defined ticker scroll speed and direction; and
an animation program module identifying any animated graphical
element mapped to said buffer which requires updating, and for each identified
animated graphical element, retrieving a respective next frame and refreshing a
corresponding present frame in the buffer with a next frame.

17. The computer program of claim 16, further comprising
a program module which, upon the detection of a condition when the
size of the ticker window exceeds that between the offset point and an end of the
buffer:
retrieves additional ticker element objects for display in said
ticker when said detected condition is true; and
maps said retrieved additional ticker element objects to said end
of the buffer.

18. The computer program of claim 16, wherein:
a particular ticker element object comprises an animated graphical
element including a plurality of predefined frames in a native file format;
the data retrieval program module further:
retrieving the predefined frames associated with animated
graphical element;
converting each of said retrieved frames to a bitmap format;
and
generating a static bitmap image comprising said retrieved
converted frames and a size data field indicating the number of frames in said static
bitmap image;
the mapping program module mapping a first frame from said
respective animated graphical element to said buffer; and
the animation program module retrieving a respective next frame for
the particular ticker element object by extracting the next frame from the respective
static bitmap image.

19. The computer program of claim 16, wherein said ticker element objects
are retrieved from a remote location.
20. The computer program of claim 16, wherein:

each ticker element object is implemented in an object oriented
programming environment and is embedded with information necessary to
autonomously generate appropriate data to be mapped to said buffer; and
the mapping program module instructing each respective ticker
element object to render itself within an indicated portion of the buffer.

21. A computer program for generating a scrolling ticker on a computer
display, the ticker comprising a ticker window being scrollable along a length, the
computer program comprising:

a program module retrieving a plurality of headline entries from at
least one selected data channel located on a remote site, each headline entry
comprising a text headline for display in said ticker and storing said retrieved headline
entries in at least one corresponding headline channel file;

a program module retrieving a plurality of non-headline entries from a
remote source, each non-headline entry comprising a banner pointer indicating a
source of at least one frame of graphical data for display in said ticker, and storing
said retrieved non-headline entries in a local ad channel file;

a program module for establishing a ticker window on said display and
allocating a buffer with a length greater than the length of the ticker;

a program module selecting at least one of a headline entry from said
headline channel files and a non-headline entry from said ad channel file for display in
said ticker;

a program module converting the respective text headline for each
selected headline entry into graphical data and storing said graphical data in said
buffer;

a program module retrieving for each selected non-headline entry at
least a first frame of the graphical data identified by the associated banner pointer and
mapping the first frame from each selected non-headline entry to said buffer;
a program module repeatedly copying a portion of said buffer
beginning at an offset point into said ticker window and repositioning said offset point
in accordance with a defined ticker scroll speed and direction to thereby produce a
scrolling display in said ticker window; and

a program module identifying selected non-headline entries associated
with a plurality of frames of graphical data and which require a frame update, and for
each identified non-headline entry, refreshing a corresponding present frame in the
buffer with a next frame.

22. The computer program of claim 21, further comprising:
a program module retrieving updated headline entries from said
selected data channels on a periodic basis; and storing said retrieved updated headline
texts said corresponding headline channel files.

23. The computer program of claim 21, wherein the program module
retrieving updated headline entries retrieves updated headline entries for a first
particular selected data channel, and pauses a predetermined period of time prior to
retrieving updated headline entries for a second particular selected data channel.
24. A computer program for displaying animated graphical elements comprising a plurality of frames in a scrolling ticker window a computer display, the ticker window having a height and length and being scrollable along its length, the computer program comprising:

a program module allocating a buffer with a size larger than the ticker size;

a program module retrieving a plurality of ticker element objects for display in said ticker;

a program module determining the orientation of said ticker, and when said ticker orientation is horizontal, mapping said retrieved ticker element objects to said buffer in a horizontal orientation; and alternatively, when said ticker orientation is vertical, separating said retrieved ticker element objects into discrete components and mapping said retrieved ticker element objects to said buffer in a vertical configuration, wherein each discrete component is mapped in a horizontal orientation;

and

a program module repeatedly copying a portion of said buffer beginning at an offset point along said length into said ticker window and repositioning said offset point in accordance with a defined ticker scroll speed and direction.

25. The computer program of claim 24, wherein:

each ticker element object represents one of a static text element, a static graphical element, and an animated graphical element;

the program further comprising a program module identifying any animated graphical element mapped to said buffer which requires updating; and, for
each identified animated graphical element, retrieving a respective next frame and refreshing a corresponding present frame in the buffer with a next frame.
Fig. 2
Fig. 3C

Fig. 3D

SUBSTITUTE SHEET (RULE 26)
**Fig. 6**

START

100. Request sufficient Ticker Element Objects to fill buffer; Store Ticker Element Objects in list

102. LOAD Ticker Element Objects into memory

104. Reset Ticker Window Offset

106. Map Ticker Element Data into Buffer

108. Starting at window offset, copy buffer data into ticker window

7/8

110. Any graphic elements need updating?

Yes

114. Update graphic element in buffer with next frame

116. Increment/Decrement window offset

No

118. Has window moved beyond buffer?

Yes

120. Request Additional Ticker Object Elements

122. Add new elements to end of list overrun by window

124. Remove Ticker Elements from opposite end of list until list fits in buffer

No
Fig. 7A

Fig. 7B

Fig. 7C

SUBSTITUTE SHEET (RULE 26)
INTERNATIONAL SEARCH REPORT

A. CLASSIFICATION OF SUBJECT MATTER
IPC(6) : G06T 15/00
US CL : 345/123
According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED
Minimum documentation searched (classification system followed by classification symbols)
U.S. : 345/123, 121, 124, 125, 473, 474

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic database consulted during the international search (name of database and, where practical, search terms used)
IEL Online
Search terms: scrol* ticker

C. DOCUMENTS CONSIDERED TO BE RELEVANT

<table>
<thead>
<tr>
<th>Category*</th>
<th>Citation of document, with indication, where appropriate, of the relevant passages</th>
<th>Relevant to claim No.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Y,E</td>
<td>US 5,959,621 A (NAWAZ et al.) 28 September 1999, whole document</td>
<td>1-25</td>
</tr>
<tr>
<td>Y,E</td>
<td>US 5,818,935 A (MAA) 06 October 1998, whole document</td>
<td>1-25</td>
</tr>
<tr>
<td>Y</td>
<td>US 5,768,528 A (STUMM) 16 June 1998, whole document</td>
<td>1-25</td>
</tr>
</tbody>
</table>

Further documents are listed in the continuation of Box C. See patent family annex.

* Special categories of cited documents:
  "A" document defining the general state of the art which is not considered to be of particular relevance
  "E" earlier document published on or after the international filing date
  "L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)
  "O" document referring to an oral disclosure, use, exhibition or other means
  "P" document published prior to the international filing date but later than the priority date claimed
  "T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
  "X" document of particular relevance, the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
  "Y" document of particular relevance, the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
  "&" document member of the same patent family

Date of the actual completion of the international search
08 NOVEMBER 1999

Date of mailing of the international search report
04 FEB 2000

Authorized officer
MANO PADMANABHAN

Name and mailing address of the ISA/US Commissioner of Patents and Trademarks
Box PCT
Washington, D.C. 20231
Facsimile No. (703) 305-3230

Telephone No. (703) 306-2903

Form PCT/ISA/210 (second sheet) (July 1992)*